Methodology for Repeater Insertion in the Itanium™ Microprocessor

R. McInerney, K. Leeper, T. Hill,
H. Chan, L. McQuiddy,
Bulent Basaran

Microprocessor Group

Intel Corporation

intel

Problem Overview

- Finally, RC delay *really* dominates gate delay!
  - Itanium™ is the first chip (at Intel) that used repeaters on global nets extensively
  - Process:
    - 0.18µ CMOS
    - 6 metal layers

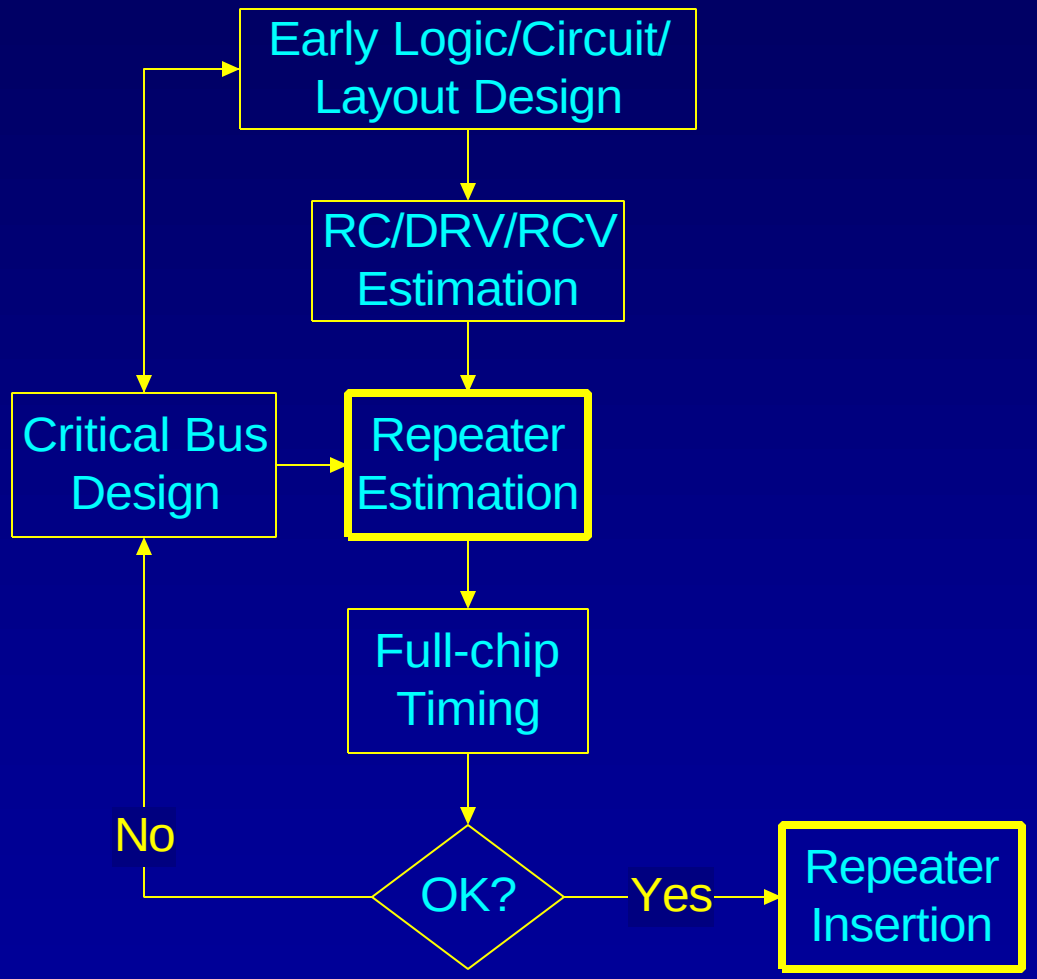
  => Requires up to 6 repeaters to span the die
- Itanium design needed a methodology and a set of supporting tools to attack this “new” problem

intel.

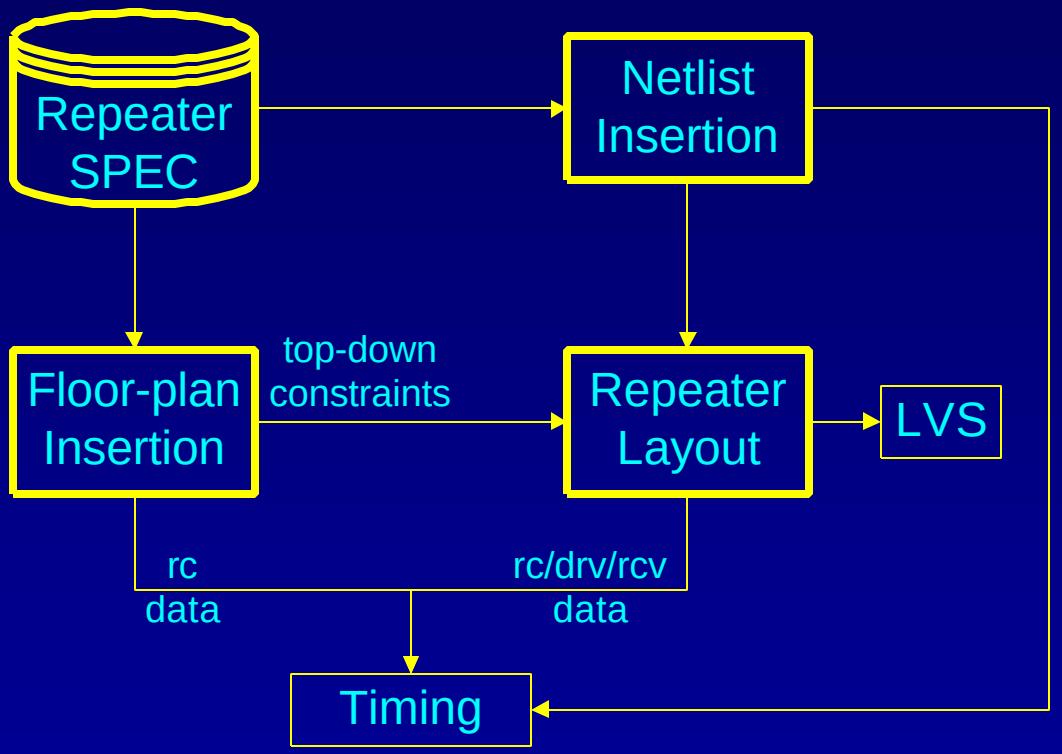
Outline

- High-level flow
- Repeater estimation
- Physical design of repeaters
- Repeaters in the netlist
- Results
- Conclusion
- Future Challenges

High-level Flow: Estimation



High-level Flow: Insertion



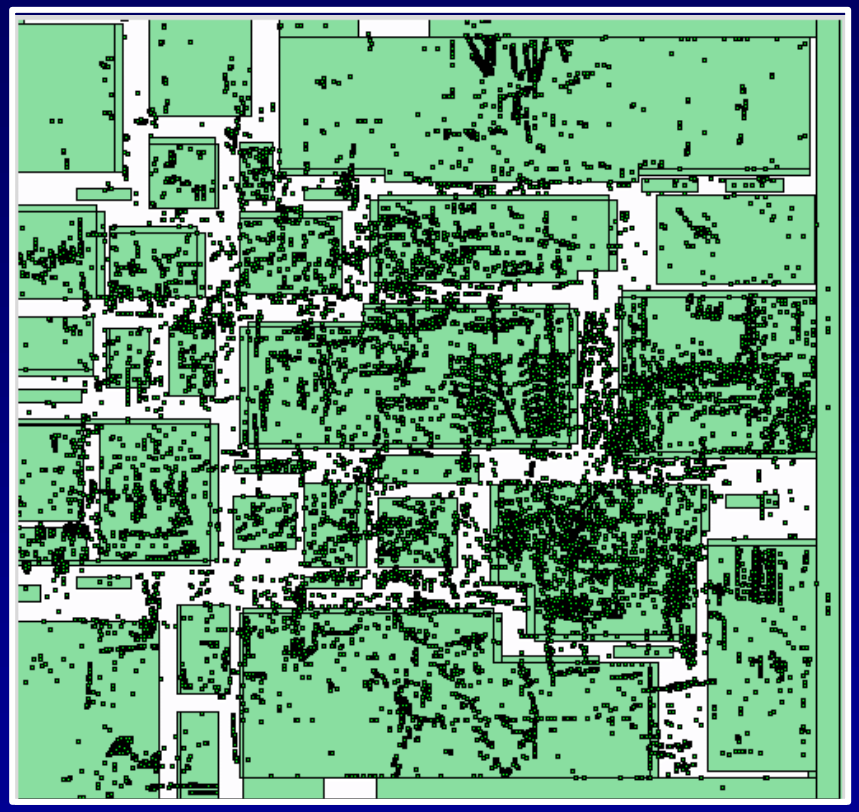
intel.

Repeater Estimation

- Two phases of operation:
  - Early: Unconstrained repeater locations
  - Late: Block/routing constraint-aware estimation
- Inputs:
  - Target slope/delay
  - Pre-characterized repeater library
  - Early phase:
    - Estimated interconnect, driver/receiver RC
  - Late phase:
    - Accurate extracted RC
    - Repeater block specifications
- Output: SPEC format (type/place/size)

intel.

Repeater Estimation - Early Phase



- Scatter plot of ideal repeater locations in the floor-plan
  - 10% guard-band on delay
  - Table look-up for location/size of repeaters
  - Estimated interconnect RC
  - Estimated driver/receiver
- Data used for:
  - Early timing analysis
  - Plan repeater blocks

intel.

Repeater Estimation - Late Phase

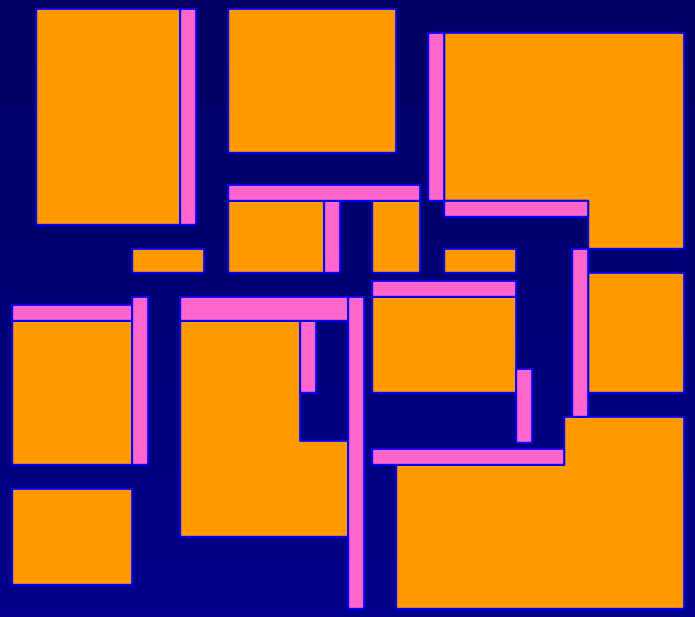
- Late phase starts after
  - Repeater blocks are planned
  - Most global nets are routed
- Repeater engine:
  - Dynamic programming for solution enumeration
  - Slope constraints to prune search tree, size repeaters
  - Best delay to prune
- Delay estimation based on AWE
- X-talk noise considered (per pin)
- Later in design, explore repeater re-sizing

Physical Design of Repeaters

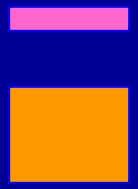
- Major design decisions:
  - Cluster all* repeaters into Repeater Stations (RPS)
    - Manage complexity
    - Simplify flow/tools
  - Create RPSs only at the full-chip level in hierarchy
    - Local nets don’t need as many repeaters
    - Pro: de-couple design of functional units from repeaters
    - Con: further away from ideal repeater location (per net)
  - Impose a standard/restricted template on RPSs
    - Reduce area/track usage
    - Enable accurate top-down planning
    - Facilitate bottom-up automation

Repeater Station Planning

- Two kinds of repeater stations:
  - vertical to repeat horizontal wires
  - horizontal to repeat vertical wires
- Thin, but maximally long (as needed)
- Trade-off between:
  - performance/area, versus
  - design structure/productivity



A section (1/5th) of Itanium™



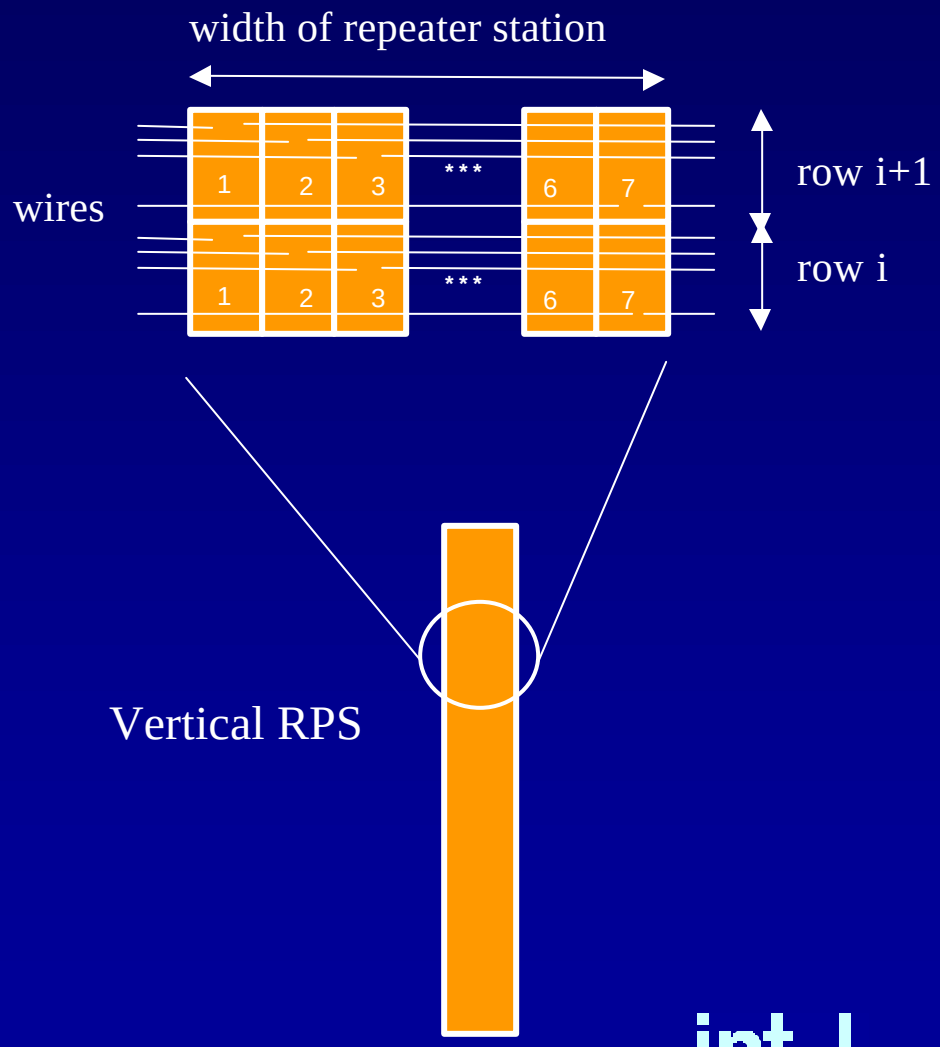
RPS

Unit

intel.

Repeater Station Design

- 2D array of repeaters:
  - Maximum 7 cells in a row
  - Can repeat all 7 tracks crossing over a row
  - Rows are stacked as needed to create a 7xN array
  - The width can be reduced to save resources when utilization is not high
- If a wire crosses an RPS, it can be repeated
- No jogging over RPS allowed



Repeater Cell Design

- Pre-characterized standard cell library contains:
  - Buffers
  - Inverters
  - Latch/flip-flops
  - and others...
- A set of driver strengths for varying net lengths
  - Reduce power, improve noise
- Layout: standard template for all cells
  - One size fits all
  - I/O, pwr/clk, control tracks pre-designed
  - Open feedthru tracks: about 50% on M2 and M3

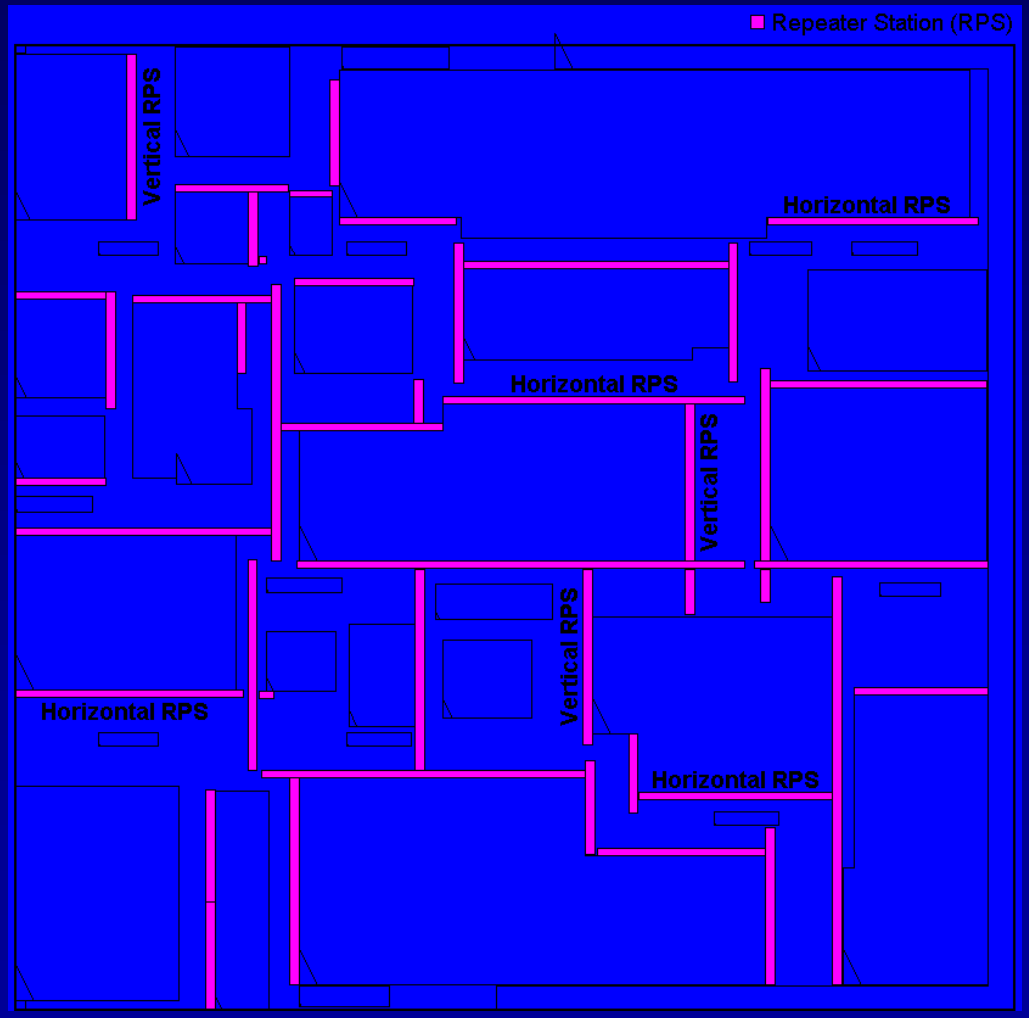
intel.

Repeaters in the Netlist

- Most design tasks (circuit/layout/timing) require the repeated netlist
  => Insert repeaters into the RTL
- Mostly automated, except:
  - Latches/flip-flops/inverters
- Use a consistent naming convention of repeated nets
  - Used across all disciplines
- Fits well into the existing flows:
  - Verification flow unchanged
- Minimal impact on RTL simulation performance

intel.

Result: Itanium™ Floor-plan with RPSs



Results

- ~80% of global routes are repeated
  - Compare with Pentium III at 5%
- ~10% used latch/FF
- ~2 repeaters per net (average)
- 45 repeater stations
- Repeater station utilization ~40%
- Power/noise impact negligible
- Met frequency goal (800 MHz) for tape-out

Conclusion

- Repeater methodology helped reduce impact on:
  - Design schedule
  - Chip area
- Estimation accuracy was very high:
  - Timing model was stable after insertion
  - Helped first silicon work at target frequency
- Most of the effort spent in *how* to insert a repeater into the design, **not** *where* or *how big*

intel.

Future Challenges

- Design a repeater grid prior to traditional floor-planning of unit blocks
- Leverage local repeater blocks
- Automate repeater insertion for local nets
- Improve router technology for repeater-awareness

intel.